

EEEEEEEEEEEEEEEE	DDDDDDDDDDDDDD	TTTTTTTTTTTTTTTT
EEEEEEEEEEEEEEEE	DDDDDDDDDDDDDD	TTTTTTTTTTTTTTTT
EEEEEEEEEEEEEEEE	DDDDDDDDDDDDDD	TTTTTTTTTTTTTTTT
EEE	DDD	TTT
EEE	DDD	TTT
EEE	DDD	TTT
EEE	DDD	TTT
EEE	DDD	TTT
EEE	DDD	TTT
EEEEEEEEEEEEEE	DDD	TTT
EEEEEEEEEEEEEE	DDD	TTT
EEEEEEEEEEEEEE	DDD	TTT
EEE	DDD	TTT
EEE	DDD	TTT
EEE	DDD	TTT
EEE	DDD	TTT
EEE	DDD	TTT
EEE	DDD	TTT
EEEEEEEEEEEEEEEE	DDDDDDDDDDDDDD	TTT
EEEEEEEEEEEEEEEE	DDDDDDDDDDDDDD	TTT
EEEEEEEEEEEEEEEE	DDDDDDDDDDDDDD	TTT

```
CCCCCCCC HH HH MM MM KK KK EEEEEEEEE YY YY WW WW RRRRRRR DDDDDDD
CCCCCCCC HH HH MM MM KK KK EEEEEEEEE YY YY WW WW RRRRRRR DDDDDDD
CC HH HH MMM MMM KK KK EE YY YY WW WW RR RR DD DD
CC HH HH MMM MMM KK KK EE YY YY WW WW RR RR DD DD
CC HH HH MM MM KK KK EE YY YY WW WW RR RR DD DD
CC HH HH MM MM KK KK EE YY YY WW WW RR RR DD DD
CC HHHHHHHH MM MM KKKKK EEEEEEE YY YY WW WW RRRRRRR DD DD
CC HHHHHHHH MM MM KKKKK EEEEEEE YY YY WW WW RRRRRRR DD DD
CC HH HH MM MM KK KK EE YY WW WW RR RR DD DD
CC HH HH MM MM KK KK EE YY WW WW RR RR DD DD
CC HH HH MM MM KK KK EE YY WWW WWW RR RR DD DD
CC HH HH MM MM KK KK EE YY WW WW RR RR DD DD
CCCCCCCC HH HH MM MM KK KK EEEEEEEEE YY WW WW RR RR DDDDDDD
CCCCCCCC HH HH MM MM KK KK EEEEEEEEE YY WW WW RR RR DDDDDDD
.....
.....
.....
.....
```

```
LL LL II II SSSSSSS
LL LL II II SSSSSSS
LL LL II II SS
LL LL II II SS
LL LL II II SS
LL LL II II SSSSSS
LL LL II II SSSSSS
LL LL II II SS
LL LL II II SS
LL LL II II SS
LLLLLLLLL II II SSSSSSS
LLLLLLLLL II II SSSSSSS
```



```
0001 0 %TITLE 'EDT$CHMKEYWRD - look for a keyword'
0002 0 MODULE EDT$CHMKEYWRD (
0003 0 IDENT = 'V04-000'
0004 0 ) =
0005 1 BEGIN
0006 1
0007 1 *****
0008 1 *
0009 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0010 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0011 1 * ALL RIGHTS RESERVED.
0012 1 *
0013 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0014 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0015 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0016 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0017 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0018 1 * TRANSFERRED.
0019 1 *
0020 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0021 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0022 1 * CORPORATION.
0023 1 *
0024 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0025 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0026 1 *
0027 1 *
0028 1 *****
0029 1
0030 1
0031 1 ++
0032 1 FACILITY: EDT -- The DEC Standard Editor
0033 1
0034 1 ABSTRACT:
0035 1
0036 1 This module compares the command buffer contents to a table
0037 1 of keywords.
0038 1
0039 1 ENVIRONMENT: Runs at any access mode - AST reentrant
0040 1
0041 1 AUTHOR: Bob Kushlis, CREATION DATE: Unknown
0042 1
0043 1 MODIFIED BY:
0044 1
0045 1 1-001 - Original. DJS 04-Feb-1981. This module was created by
0046 1 extracting the routine EDT$KWORD from module CHANGE.BLI.
0047 1 1-002 - Regularize headers. JBS 03-Mar-1981
0048 1 1-004 - Change to a table arranged alphabetically. STS 21-Sep-1982
0049 1 1-005 - Move the keywords here from EDT$CHMPARSE, to reduce the program
0050 1 size on the PDP-11. Also, put an underscore in the entry point name. JBS 29-Sep-1982
0051 1 1-006 - Accept lower case letters as equivalent to upper case, and improve error
0052 1 checking. JBS 01-Oct-1982
0053 1 1-007 - Make this routine position-independent. JBS 01-Oct-1982
0054 1 1-008 - Add conditionals for WPS and VT220. JBS 10-Feb-1983
0055 1 1-009 - Don't forget the SUPPORTS library. JBS 11-Feb-1983
0056 1
0057 1 --
```



```
: 59      0058 1 %SBTTL 'Declarations'
: 60      0059 1
: 61      0060 1 | TABLE OF CONTENTS:
: 62      0061 1 |
: 63      0062 1 |
: 64      0063 1 REQUIRE 'EDT$SRC:TRAROUNAM';
: 65      0502 1
: 66      0503 1 FORWARD ROUTINE
: 67      0504 1     EDT$KEY_WORD : NOVALUE;
: 68      0505 1
: 69      0506 1 |
: 70      0507 1 | INCLUDE FILES:
: 71      0508 1 |
: 72      0509 1 |
: 73      0510 1 REQUIRE 'EDT$SRC:EDTREQ';
: 74      0645 1
: 75      0646 1 LIBRARY 'EDT$SRC:TRANSLATE';
: 76      0647 1
: 77      0648 1 LIBRARY 'EDT$SRC:SUPPORTS';
: 78      0649 1
: 79      0650 1 |
: 80      0651 1 | MACROS:
: 81      0652 1 |
: 82      0653 1 |     NONE
: 83      0654 1 |
: 84      0655 1 | EQUATED SYMBOLS:
: 85      0656 1 |
: 86      0657 1 |
: 87      0658 1 FIELD
: 88      0659 1     KEY_WORD_FIELD =
: 89      0660 1     SET
: 90      0661 1     KEY_WORD_NEXT = [0, 0, %BPADDR, 0],
: 91      0662 1     KEY_WORD_NUM = [%BPADDR/8, 0, 8, 0],
: 92      0663 1     KEY_WORD_LEN = [(%BPADDR/8), 8, 8, 0],
: 93      0664 1     KEY_WORD_POINTER = [(%BPADDR/8) + 2, 0, %BPADDR, 0]
: 94      0665 1     TES;
: 95      0666 1
: 96      0667 1 |
: 97      0668 1 | OWN STORAGE:
: 98      0669 1 |
: 99      0670 1 | +
100      0671 1 | Define the keywords used to make up change mode sub-commands.
101      0672 1 |
102      0673 1 | Each record in this table contains a address pointer to the next keyword
103      0674 1 | with this alphabetic character the value of the keyword, its length,
104      0675 1 | and the ASCII characters which comprise it.
105      0676 1 |
106      0677 1 | The table is a concatenation of keyword entries. Each
107      0678 1 | consists of a pointer to the next keyword to examine if
108      0679 1 | this one should fail, a keyword number byte, length byte,
109      0680 1 | and the ASCII text for the keyword. A 0 length byte
110      0681 1 | marks the end of the table. Letters in keywords are
111      0682 1 | all upper case.
112      0683 1 | -
113      0684 1 |
114      0685 1 BIND
115      0686 1     ADDR_BASE = UPLIT (0),
```

! Compare the command buffer contents to a table of keywords


```
116 0687 1 END_VERBS = UPLIT (
117 0688 1 0, BYTE(0), BYTE(0), 0),
118 0689 1 ASC_VERB = UPLIT (
119 0690 1 END_VERBS - ADDR_BASE, BYTE(VERB_K_ASC), BYTE(3), BYTE('ASC')),
120 0691 1 ADV_VERB = UPLIT (
121 0692 1 ASC_VERB - ADDR_BASE, BYTE(VERB_K_ADV), BYTE(3), BYTE('ADV')),
122 0693 1 A_VERBS = UPLIT (
123 0694 1 ADV_VERB - ADDR_BASE, BYTE(VERB_K_APPEND), BYTE(6), BYTE('APPEND')),
124 0695 1 BELL_VERB = UPLIT (
125 0696 1 END_VERBS - ADDR_BASE, BYTE(VERB_K_BELL), BYTE(4), BYTE('BELL')),
126 0697 1 B_VERBS = UPLIT (
127 0698 1 BELL_VERB - ADDR_BASE, BYTE(VERB_K_BACK), BYTE(4), BYTE('BACK')),
128 0699 1 CUT_VERB = UPLIT (
129 0700 1 END_VERBS - ADDR_BASE, BYTE(VERB_K_CUT), BYTE(3), BYTE('CUT')),
130 0701 1 CLSS_VERB = UPLIT (
131 0702 1 CUT_VERB - ADDR_BASE, BYTE(VERB_K_CLSS), BYTE(4), BYTE('CLSS')),
132 0703 1 CHGL_VERB = UPLIT (
133 0704 1 CLSS_VERB - ADDR_BASE, BYTE(VERB_K_CHGL), BYTE(4), BYTE('CHGL')),
134 0705 1 CHGU_VERB = UPLIT (
135 0706 1 CHGL_VERB - ADDR_BASE, BYTE(VERB_K_CHGU), BYTE(4), BYTE('CHGU')),
136 0707 1 C_VERBS = UPLIT (
137 0708 1 CHGU_VERB - ADDR_BASE, BYTE(VERB_K_CHGC), BYTE(4), BYTE('CHGC')),
138 0709 1 D_VERB = UPLIT (
139 0710 1 END_VERBS - ADDR_BASE, BYTE(VERB_K_DELETE), BYTE(1), BYTE('D')),
140 0711 1 DUPLC_VERB = UPLIT (
141 0712 1 D_VERB - ADDR_BASE, BYTE(VERB_K_DUPLC), BYTE(4), BYTE('DUPLC')),
142 0713 1 DMOV_VERB = UPLIT (
143 0714 1 DUPLC_VERB - ADDR_BASE, BYTE(VERB_K_DMOV), BYTE(4), BYTE('DMOV')),
144 0715 1 DLWC_VERB = UPLIT (
145 0716 1 DMOV_VERB - ADDR_BASE, BYTE(VERB_K_DLWC), BYTE(4), BYTE('DLWC')),
146 0717 1 DEFK_VERB = UPLIT (
147 0718 1 DLWC_VERB - ADDR_BASE, BYTE(VERB_K_DEFK), BYTE(4), BYTE('DEFK')),
148 0719 1 DATE_VERB = UPLIT (
149 0720 1 DEFK_VERB - ADDR_BASE, BYTE(VERB_K_DATE), BYTE(4), BYTE('DATE')),
150 0721 1 D_VERBS = UPLIT (
151 0722 1 DATE_VERB - ADDR_BASE, BYTE(VERB_K_DESEL), BYTE(5), BYTE('DESEL')),
152 0723 1 EX_VERB = UPLIT (
153 0724 1 END_VERBS - ADDR_BASE, BYTE(VERB_K_EXIT), BYTE(2), BYTE('EX')),
154 0725 1 E_VERBS = UPLIT (
155 0726 1 EX_VERB - ADDR_BASE, BYTE(VERB_K_EXT), BYTE(3), BYTE('EXT')),
156 0727 1 F_VERBS = UPLIT (
157 0728 1 END_VERBS - ADDR_BASE, BYTE(VERB_K_FILL), BYTE(4), BYTE('FILL')),
158 0729 1 H_VERBS = UPLIT (
159 0730 1 END_VERBS - ADDR_BASE, BYTE(VERB_K_HELP), BYTE(4), BYTE('HELP')),
160 0731 1 I_VERBS = UPLIT (
161 0732 1 END_VERBS - ADDR_BASE, BYTE(VERB_K_INSERT), BYTE(1), BYTE('I')),
162 0733 1 K_VERBS = UPLIT (
163 0734 1 END_VERBS - ADDR_BASE, BYTE(VERB_K_KS), BYTE(2), BYTE('KS')),
164 0735 1 P_VERBS = UPLIT (
165 0736 1 END_VERBS - ADDR_BASE, BYTE(VERB_K_PASTE), BYTE(5), BYTE('PASTE')),
166 0737 1 Q_VERBS = UPLIT (
167 0738 1 END_VERBS - ADDR_BASE, BYTE(VERB_K_QUIT), BYTE(4), BYTE('QUIT')),
168 0739 1 R_VERB = UPLIT (
169 0740 1 END_VERBS - ADDR_BASE, BYTE(VERB_K_REPLACE), BYTE(1), BYTE('R')),
170 0741 1 R_VERBS = UPLIT (
171 0742 1 R_VERB - ADDR_BASE, BYTE(VERB_K_REF), BYTE(3), BYTE('REF')),
172 0743 1 S_VERB = UPLIT (
```



```
173 0744 1      END_VERBS - ADDR_BASE, BYTE(VERB_K_SUBS),    BYTE(1),    BYTE('S')),
174 0745 1      SN_VERB = UPLIT (
175 0746 1      S_VERB - ADDR_BASE,    BYTE(VERB_K_SN),    BYTE(2),    BYTE('SN')),
176 0747 1      SHR_VERB = UPLIT (
177 0748 1      SN_VERB - ADDR_BASE,    BYTE(VERB_K_SHR),    BYTE(3),    BYTE('SHR')),
178 0749 1      SHL_VERB = UPLIT (
179 0750 1      SHR_VERB - ADDR_BASE,    BYTE(VERB_K_SHL),    BYTE(3),    BYTE('SHL')),
180 0751 1      SEL_VERB = UPLIT (
181 0752 1      SHL_VERB - ADDR_BASE,    BYTE(VERB_K_SEL),    BYTE(3),    BYTE('SEL')),
182 0753 1      S_VERBS = UPLIT (
183 0754 1      SEL_VERB - ADDR_BASE,    BYTE(VERB_K_SSEL),    BYTE(4),    BYTE('SSEL')),
184 0755 1      TI_VERB = UPLIT (
185 0756 1      END_VERBS - ADDR_BASE,    BYTE(VERB_K_TI),    BYTE(2),    BYTE('TI')),
186 0757 1      TD_VERB = UPLIT (
187 0758 1      TI_VERB - ADDR_BASE,    BYTE(VERB_K_TD),    BYTE(2),    BYTE('TD')),
188 0759 1      TC_VERB = UPLIT (
189 0760 1      TD_VERB - ADDR_BASE,    BYTE(VERB_K_TC),    BYTE(2),    BYTE('TC')),
190 0761 1      TOP_VERB = UPLIT (
191 0762 1      TC_VERB - ADDR_BASE,    BYTE(VERB_K_TOP),    BYTE(3),    BYTE('TOP')),
192 0763 1      TAB_VERB = UPLIT (
193 0764 1      TOP_VERB - ADDR_BASE,    BYTE(VERB_K_TAB),    BYTE(3),    BYTE('TAB')),
194 0765 1      TADJ_VERB = UPLIT (
195 0766 1      TAB_VERB - ADDR_BASE,    BYTE(VERB_K_TADJ),    BYTE(4),    BYTE('TADJ')),
196 0767 1      T_VERBS = UPLIT (
197 0768 1      TADJ_VERB - ADDR_BASE,    BYTE(VERB_K_TGSEL),    BYTE(5),    BYTE('TGSEL')),
198 0769 1      UNDW_VERB = UPLIT (
199 0770 1      END_VERBS - ADDR_BASE,    BYTE(VERB_K_UNDW),    BYTE(4),    BYTE('UNDW')),
200 0771 1      UNDL_VERB = UPLIT (
201 0772 1      UNDW_VERB - ADDR_BASE,    BYTE(VERB_K_UNDL),    BYTE(4),    BYTE('UNDL')),
202 0773 1      U_VERBS = UPLIT (
203 0774 1      UNDL_VERB - ADDR_BASE,    BYTE(VERB_K_UNDC),    BYTE(4),    BYTE('UNDC')),
204 0775 1      X_VERBS = UPLIT (
205 0776 1      END_VERBS - ADDR_BASE,    BYTE(VERB_K_XLATE),    BYTE(5),    BYTE('XLATE')),
206 0777 1      CARET_VERB = UPLIT (
207 0778 1      END_VERBS - ADDR_BASE,    BYTE(VERB_K_CC),    BYTE(1),    BYTE('^'));
208 0779 1
209 0780 1  BIND
210 0781 1      VERB_TABLE = UPLIT (
211 0782 1      A_VERBS - ADDR_BASE,
212 0783 1      B_VERBS - ADDR_BASE,
213 0784 1      C_VERBS - ADDR_BASE,
214 0785 1      D_VERBS - ADDR_BASE,
215 0786 1      E_VERBS - ADDR_BASE,
216 0787 1      F_VERBS - ADDR_BASE,
217 0788 1      END_VERBS - ADDR_BASE,
218 0789 1      H_VERBS - ADDR_BASE,
219 0790 1      I_VERBS - ADDR_BASE,
220 0791 1      END_VERBS - ADDR_BASE,
221 0792 1      K_VERBS - ADDR_BASE,
222 0793 1      END_VERBS - ADDR_BASE,
223 0794 1      END_VERBS - ADDR_BASE,
224 0795 1      END_VERBS - ADDR_BASE,
225 0796 1      END_VERBS - ADDR_BASE,
226 0797 1      P_VERBS - ADDR_BASE,
227 0798 1      Q_VERBS - ADDR_BASE,
228 0799 1      R_VERBS - ADDR_BASE,
229 0800 1      S_VERBS - ADDR_BASE,
```



```
230      0801 1      T_VERBS - ADDR_BASE,
231      0802 1      U_VERBS - ADDR_BASE,
232      0803 1      END_VERBS - ADDR_BASE,
233      0804 1      END_VERBS - ADDR_BASE,
234      0805 1      X_VERBS - ADDR_BASE,
235      0806 1      END_VERBS - ADDR_BASE,
236      0807 1      END_VERBS - ADDR_BASE,
237      0808 1      END_VERBS - ADDR_BASE,
238      0809 1      END_VERBS - ADDR_BASE,
239      0810 1      END_VERBS - ADDR_BASE,
240      0811 1      CARET_VERB - ADDR_BASE);
241      0812 1
242      0813 1
243      0814 1      !+ The following are keywords which are entities.
244      0815 1
245      0816 1      !- The values must be separated by two so we can add the direction to
246      0817 1      !- the entity for use as a case index.
247      0818 1
248      0819 1
249      0820 1 BIND
250      0821 1      END_ENTITY = UPLIT (
251      0822 1      0, BYTE(0), BYTE(0), 0),
252      0823 1      BW_ENT = UPLIT (
253      0824 1      END_ENTITY - ADDR_BASE, BYTE(ENT_K_BW), BYTE(2), BYTE('BW')),
254      0825 1      BR_ENT = UPLIT (
255      0826 1      BW_ENT - ADDR_BASE, BYTE(ENT_K_BR), BYTE(2), BYTE('BR')),
256      0827 1      BL_ENT = UPLIT (
257      0828 1      BR_ENT - ADDR_BASE, BYTE(ENT_K_BL), BYTE(2), BYTE('BL')),
258      0829 1      BPAR_ENT = UPLIT (
259      0830 1      BL_ENT - ADDR_BASE, BYTE(ENT_K_BPAR), BYTE(4), BYTE('BPAR')),
260      0831 1      BSEN_ENT = UPLIT (
261      0832 1      BPAR_ENT - ADDR_BASE, BYTE(ENT_K_BSEN), BYTE(4), BYTE('BSEN')),
262      0833 1      B_ENTS = UPLIT (
263      0834 1      BSEN_ENT - ADDR_BASE, BYTE(ENT_K_BPAGE), BYTE(5), BYTE('BPAGE')),
264      0835 1      C_ENTS = UPLIT (
265      0836 1      END_ENTITY - ADDR_BASE, BYTE(ENT_K_CHAR), BYTE(1), BYTE('C')),
266      0837 1      EW_ENT = UPLIT (
267      0838 1      END_ENTITY - ADDR_BASE, BYTE(ENT_K_EW), BYTE(2), BYTE('EW')),
268      0839 1      EL_ENT = UPLIT (
269      0840 1      EW_ENT - ADDR_BASE, BYTE(ENT_K_EL), BYTE(2), BYTE('EL')),
270      0841 1      ER_ENT = UPLIT (
271      0842 1      EL_ENT - ADDR_BASE, BYTE(ENT_K_ER), BYTE(2), BYTE('ER')),
272      0843 1      EPAR_ENT = UPLIT (
273      0844 1      ER_ENT - ADDR_BASE, BYTE(ENT_K_EPAR), BYTE(4), BYTE('EPAR')),
274      0845 1      ESEN_ENT = UPLIT (
275      0846 1      EPAR_ENT - ADDR_BASE, BYTE(ENT_K_ESEN), BYTE(4), BYTE('ESEN')),
276      0847 1      E_ENTS = UPLIT (
277      0848 1      ESEN_ENT - ADDR_BASE, BYTE(ENT_K_EPAGE), BYTE(5), BYTE('EPAGE')),
278      0849 1      L_ENTS = UPLIT (
279      0850 1      END_ENTITY - ADDR_BASE, BYTE(ENT_K_LINE), BYTE(1), BYTE('L')),
280      0851 1      N_ENTS = UPLIT (
281      0852 1      END_ENTITY - ADDR_BASE, BYTE(ENT_K_NL), BYTE(2), BYTE('NL')),
282      0853 1      PAR_ENT = UPLIT (
283      0854 1      END_ENTITY - ADDR_BASE, BYTE(ENT_K_PAR), BYTE(3), BYTE('PAR')),
284      0855 1      P_ENTS = UPLIT (
285      0856 1      PAR_ENT - ADDR_BASE, BYTE(ENT_K_PAGE), BYTE(4), BYTE('PAGE')),
286      0857 1      SR_ENT = UPLIT (
```

```
287 0858 1      END_ENTITY - ADDR_BASE, BYTE(ENT_K_SR),      BYTE(2),      BYTE('SR')),
288 0859 1      S_ENTS = UPLIT (
289 0860 1      SR_ENT - ADDR_BASE,      BYTE(ENT_K_SEN),      BYTE(3),      BYTE('SEN')),
290 0861 1      V_ENTS = UPLIT (
291 0862 1      END_ENTITY - ADDR_BASE, BYTE(ENT_K_VERT),      BYTE(1),      BYTE('V')),
292 0863 1      W_ENTS = UPLIT (
293 0864 1      END_ENTITY - ADDR_BASE, BYTE(ENT_K_WORD),      BYTE(1),      BYTE('W'));
294 0865 1
295 0866 1 BIND
296 0867 1      ENTITY_TABLE = UPLIT (
297 0868 1      B_ENTS - ADDR_BASE,
298 0869 1      C_ENTS - ADDR_BASE,
299 0870 1      END_ENTITY - ADDR_BASE,
300 0871 1      E_ENTS - ADDR_BASE,
301 0872 1      END_ENTITY - ADDR_BASE,
302 0873 1      END_ENTITY - ADDR_BASE,
303 0874 1      END_ENTITY - ADDR_BASE,
304 0875 1      END_ENTITY - ADDR_BASE,
305 0876 1      END_ENTITY - ADDR_BASE,
306 0877 1      END_ENTITY - ADDR_BASE,
307 0878 1      L_ENTS - ADDR_BASE,
308 0879 1      END_ENTITY - ADDR_BASE,
309 0880 1      N_ENTS - ADDR_BASE,
310 0881 1      END_ENTITY - ADDR_BASE,
311 0882 1      P_ENTS - ADDR_BASE,
312 0883 1      END_ENTITY - ADDR_BASE,
313 0884 1      END_ENTITY - ADDR_BASE,
314 0885 1      S_ENTS - ADDR_BASE,
315 0886 1      END_ENTITY - ADDR_BASE,
316 0887 1      END_ENTITY - ADDR_BASE,
317 0888 1      V_ENTS - ADDR_BASE,
318 0889 1      W_ENTS - ADDR_BASE);
319 0890 1
320 0891 1
321 0892 1      EXTERNAL REFERENCES:
322 0893 1
323 0894 1      In the routine
```



```
: 325 0895 1 %SBTTL 'EDT$$KEY_WORD - look for a key word'
: 326 0896 1
: 327 0897 1 GLOBAL ROUTINE EDT$$KEY_WORD (      ! Look for a key word
: 328 0898 1     TABLE_NO,                    ! 1 = verb table, 2 = entity table
: 329 0899 1     KEY_NUM                      ! Key number
: 330 0900 1     ) : NOVALUE =
: 331 0901 1
: 332 0902 1 ++
: 333 0903 1 FUNCTIONAL DESCRIPTION:
: 334 0904 1
: 335 0905 1     This routine scans a table of keywords, attempting to find a match
: 336 0906 1     in the current command buffer pointed to by EDT$$A_CMD_BUF.
: 337 0907 1     The comparison is case blind.
: 338 0908 1
: 339 0909 1 FORMAL PARAMETERS:
: 340 0910 1
: 341 0911 1     TABLE_NO                    The number of the keyword table to use. 1 = use the
: 342 0912 1                                     verb table, 2 = use the entity table.
: 343 0913 1
: 344 0914 1     KEY_NUM                      The returned value for the number of the entity or
: 345 0915 1                                     verb which matched from the table. Zero indicates
: 346 0916 1                                     no match.
: 347 0917 1
: 348 0918 1 IMPLICIT INPUTS:
: 349 0919 1
: 350 0920 1     EDT$$A_CMD_END
: 351 0921 1     EDT$$A_CMD_BUF
: 352 0922 1
: 353 0923 1 IMPLICIT OUTPUTS:
: 354 0924 1
: 355 0925 1     EDT$$A_CMD_BUF
: 356 0926 1
: 357 0927 1 ROUTINE VALUE:
: 358 0928 1
: 359 0929 1     NONE
: 360 0930 1
: 361 0931 1 SIDE EFFECTS:
: 362 0932 1
: 363 0933 1     NONE
: 364 0934 1
: 365 0935 1 --
: 366 0936 1
: 367 0937 2 BEGIN
: 368 0938 2
: 369 0939 2 EXTERNAL
: 370 0940 2
: 371 L 0941 2 %IF SUPPORT_VT220
: 372 0942 2 %THEN
: 373 0943 2     EDT$$B_CHAR_INFO : BLOCKVECTOR [256, 1, BYTE], ! Information about each character
: 374 0944 2 %FI
: 375 0945 2
: 376 0946 2     EDT$$A_CMD_END,                ! End of command pointer
: 377 0947 2     EDT$$A_CMD_BUF;              ! Command string pointer
: 378 0948 2
: 379 0949 2 LOCAL
: 380 0950 2     KW_POINTER,
: 381 0951 2     FIRST_CHAR,
```

```
382      0952 2      TABLE_OFFSET,  
383      0953 2      FOUND;  
384      0954 2      TABLE;  
385      0955 2      TABLE_PTR : REF BLOCK [, BYTE] FIELD (KEY_WORD_FIELD),  
386      0956 2      C_POINTER;  
387      0957 2  
388      0958 2      .KEY_NUM = 0;  
389      0959 2      C_POINTER = .EDT$A_CMD_BUF;  
390      0960 2      FIRST_CHAR = CH$RCHAR_A(C_POINTER);  
391      0961 2  
392      L 0962 2      %IF SUPPORT_VT220  
393      0963 2      %THEN  
394      0964 2  
395      0965 2      IF .EDT$B_CHAR_INFO [.FIRST_CHAR, 0, 0, 1, 0] THEN FIRST_CHAR = .FIRST_CHAR - %C'a' + %C'A';  
396      0966 2  
397      U 0967 2      %ELSE  
398      0968 2  
399      U 0969 2      IF ((.FIRST_CHAR GEQ %C'a') AND (.FIRST_CHAR LEQ %C'z')) THEN FIRST_CHAR = .FIRST_CHAR - %C'a' + %C'A';  
400      U 0970 2  
401      0971 2      %FI  
402      0972 2  
403      0973 2      CASE .TABLE_NO FROM 1 TO 2 OF  
404      0974 2      SET  
405      0975 2  
406      0976 2      [1] :  
407      0977 2      BEGIN  
408      0978 2  
409      0979 2      IF ((.FIRST_CHAR LSS %C'A') OR (.FIRST_CHAR GTR %C'^')) THEN RETURN;  
410      0980 2  
411      0981 2      TABLE = VERB_TABLE;  
412      0982 2      TABLE_OFFSET = (.FIRST_CHAR - %C'A')*(%BPADDR/8);  
413      0983 2      TABLE_PTR = (.TABLE + .TABLE_OFFSET) + ADDR_BASE;  
414      0984 2      END;  
415      0985 2  
416      0986 2      [2] :  
417      0987 2      BEGIN  
418      0988 2  
419      0989 2      IF ((.FIRST_CHAR LSS %C'B') OR (.FIRST_CHAR GTR %C'W')) THEN RETURN;  
420      0990 2  
421      0991 2      TABLE = ENTITY_TABLE;  
422      0992 2      TABLE_OFFSET = (.FIRST_CHAR - %C'B')*(%BPADDR/8);  
423      0993 2      TABLE_PTR = (.TABLE + .TABLE_OFFSET) + ADDR_BASE;  
424      0994 2      END;  
425      0995 2  
426      0996 2      [OUTRANGE] :  
427      0997 2      ASSERT (0);  
428      0998 2      TES;  
429      0999 2  
430      1000 2      WHILE (.TABLE_PTR [KEY_WORD_LEN] NEQ 0) DO  
431      1001 2      BEGIN  
432      1002 2      KW_POINTER = TABLE_PTR [KEY_WORD_POINTER];  
433      1003 2      C_POINTER = .EDT$A_CMD_BUF;  
434      1004 2      FOUND = 1;  
435      1005 2  
436      1006 2      IF CH$PTR_LEQ (CH$PLUS (.C_POINTER, .TABLE_PTR [KEY_WORD_LEN]), .EDT$A_CMD_END)  
437      1007 2      THEN  
438      1008 2      BEGIN
```



```

: 439      1009  4
: 440      1010  4          INCR I FROM 1 TO .TABLE_PTR [KEY_WORD_LEN] DO
: 441      1011  5          BEGIN
: 442      1012  5
: 443      1013  5          LOCAL
: 444      1014  5          CHAR;
: 445      1015  5
: 446      1016  5          CHAR = CH$RCHAR_A (C_POINTER);
: 447      1017  5
: 448      L 1018  5      %IF SUPPORT_VT220
: 449      1019  5      %THEN
: 450      1020  5
: 451      1021  5          IF .EDT$$B_CHAR_INFO [.CHAR, 0, 0, 1, 0] THEN CHAR = .CHAR - %C'a' + %C'A';
: 452      1022  5
: 453      1023  5      %ELSE
: 454      1024  5
: 455      1025  5          IF ((.CHAR GEQ %C'a') AND (.CHAR LEQ %C'z')) THEN CHAR = .CHAR - %C'a' + %C'A';
: 456      1026  5
: 457      1027  5      %FI
: 458      1028  5
: 459      1029  5          IF (.CHAR NEQ CH$RCHAR_A (KW_POINTER)) THEN FOUND = 0;
: 460      1030  5
: 461      1031  4          END;
: 462      1032  4
: 463      1033  4          IF .FOUND
: 464      1034  4          THEN
: 465      1035  5              BEGIN
: 466      1036  5                  .KEY_NUM = .TABLE_PTR [KEY_WORD_NUM];
: 467      1037  5          !+
: 468      1038  5          !- Skip over the keyword.
: 469      1039  5
: 470      1040  5          EDT$$A_CMD_BUF = CH$PLUS (.EDT$$A_CMD_BUF, .TABLE_PTR [KEY_WORD_LEN]);
: 471      1041  5          RETURN;
: 472      1042  4          END;
: 473      1043  4
: 474      1044  3          END;
: 475      1045  3
: 476      1046  3          TABLE_PTR = .TABLE_PTR [KEY_WORD_NEXT] + ADDR_BASE;
: 477      1047  2          END;
: 478      1048  2
: 479      1049  2          RETURN;
: 480      1050  1          END;

```

! of routine EDT\$\$KEY_WORD

```

.TITLE EDT$CHMKEYWRD EDT$CHMKEYWRD - look for a keywor
.IDENT \V04-000\
.PSECT _EDT$CODE,NOWRT, SHR, PIC,2

```

```

00000000 00000 P.AAA: .LONG 0
00000000 00004 P.AAB: .LONG 0
          00 00008 .BYTE 0
          00 00009 .BYTE 0
00000000 0000A .LONG 0
          0000E .BLKB 2
00000004 00010 P.AAC: .LONG 4

```

EDT\$CHMKEYWRD
V04-000

EDT\$CHMKEYWRD - look for a keyword
EDT\$\$KEY_WORD - look for a key word

J 1
16-Sep-1984 00:03:17
14-Sep-1984 12:22:37

VAX-11 Bliss-32 V4.0-742
[EDT.SRC]CHMKEYWRD.BLI;1

Page 10
(3)

			1B	00014	.BYTE	27		
			03	00015	.BYTE	3		
	43	53	41	00016	.ASCII	\ASC\		
				00019	.BLKB	3		
			00000010	0001C	P.AAD:	.LONG	16	
			16	00020	.BYTE	22		
			03	00021	.BYTE	3		
	56	44	41	00022	.ASCII	\ADV\		
				00025	.BLKB	3		
			0000001C	00028	P.AAE:	.LONG	28	
			0A	0002C	.BYTE	10		
			06	0002D	.BYTE	6		
44	4E	45	50	50	41	0002E	.ASCII	\APPEND\
			00000004	00034	P.AAF:	.LONG	4	
			26	00038	.BYTE	38		
			04	00039	.BYTE	4		
	4C	4C	45	42	0003A	.ASCII	\BELL\	
				0003E	.BLKB	2		
			00000034	00040	P.AAG:	.LONG	52	
			17	00044	.BYTE	23		
			04	00045	.BYTE	4		
	4B	43	41	42	00046	.ASCII	\BACK\	
				0004A	.BLKB	2		
			00000004	0004C	P.AAH:	.LONG	4	
			09	00050	.BYTE	9		
			03	00051	.BYTE	3		
	54	55	43	00052	.ASCII	\CUT\		
				00055	.BLKB	3		
			0000004C	00058	P.AAI:	.LONG	76	
			2D	0005C	.BYTE	45		
			04	0005D	.BYTE	4		
	53	53	4C	43	0005E	.ASCII	\CLSS\	
				00062	.BLKB	2		
			00000058	00064	P.AAJ:	.LONG	88	
			05	00068	.BYTE	5		
			04	00069	.BYTE	4		
	4C	47	48	43	0006A	.ASCII	\CHGL\	
				0006E	.BLKB	2		
			00000064	00070	P.AAK:	.LONG	100	
			04	00074	.BYTE	4		
			04	00075	.BYTE	4		
	55	47	48	43	00076	.ASCII	\CHGU\	
				0007A	.BLKB	2		
			00000070	0007C	P.AAL:	.LONG	112	
			03	00080	.BYTE	3		
			04	00081	.BYTE	4		
	43	47	48	43	00082	.ASCII	\CHGC\	
				00086	.BLKB	2		
			00000004	00088	P.AAM:	.LONG	4	
			01	0008C	.BYTE	1		
			01	0008D	.BYTE	1		
			44	0008E	.ASCII	\D\		
				0008F	.BLKB	1		
			00000088	00090	P.AAN:	.LONG	136	
			28	00094	.BYTE	40		
			04	00095	.BYTE	4		
	43	50	55	44	00096	.ASCII	\DUPC\	

		0009A		.BLKB	2
	00000090	0009C	P.AAO:	.LONG	144
	2A	000A0		.BYTE	42
	04	000A1		.BYTE	4
56	4F 4D 44	000A2		.ASCII	\DMOV\
		000A6		.BLKB	2
	0000009C	000A8	P.AAP:	.LONG	156
	29	000AC		.BYTE	41
	04	000AD		.BYTE	4
43	57 4C 44	000AE		.ASCII	\DLWC\
		000B2		.BLKB	2
	000000A8	000B4	P.AAQ:	.LONG	168
	25	000B8		.BYTE	37
	04	000B9		.BYTE	4
4B	46 45 44	000BA		.ASCII	\DEFK\
		000BE		.BLKB	2
	000000B4	000C0	P.AAR:	.LONG	180
	27	000C4		.BYTE	39
	04	000C5		.BYTE	4
45	54 41 44	000C6		.ASCII	\DATE\
		000CA		.BLKB	2
	000000C0	000CC	P.AAS:	.LONG	192
	2B	000D0		.BYTE	43
	05	000D1		.BYTE	5
4C	45 53 45 44	000D2		.ASCII	\DESEL\
		000D7		.BLKB	1
	00000004	000D8	P.AAT:	.LONG	4
	11	000DC		.BYTE	17
	02	000DD		.BYTE	2
	58 45	000DE		.ASCII	\EX\
	000000D8	000E0	P.AAU:	.LONG	216
	23	000E4		.BYTE	35
	03	000E5		.BYTE	3
54	58 45	000E6		.ASCII	\EXT\
		000E9		.BLKB	3
	00000004	000EC	P.AAV:	.LONG	4
	07	000F0		.BYTE	7
	04	000F1		.BYTE	4
4C	4C 49 46	000F2		.ASCII	\FILL\
		000F6		.BLKB	2
	00000004	000F8	P.AAW:	.LONG	4
	1A	000FC		.BYTE	26
	04	000FD		.BYTE	4
50	4C 45 48	000FE		.ASCII	\HELP\
		00102		.BLKB	2
	00000004	00104	P.AAX:	.LONG	4
	0E	00108		.BYTE	14
	01	00109		.BYTE	1
	49	0010A		.ASCII	\I\
		0010B		.BLKB	1
	00000004	0010C	P.AAY:	.LONG	4
	24	00110		.BYTE	36
	02	00111		.BYTE	2
	53 4B	00112		.ASCII	\KS\
	00000004	00114	P.AAZ:	.LONG	4
	0D	00118		.BYTE	13
	05	00119		.BYTE	5

EDT\$CHMKEYWRD
V04-000

EDT\$CHMKEYWRD - look for a keyword
EDT\$\$KEY_WORD - look for a key word

L 1
16-Sep-1984 00:03:17
14-Sep-1984 12:22:37

VAX-11 Bliss-32 V4.0-742
[EDT.SRC]CHMKEYWRD.BLI;1

Page 12
(3)

45	54	53	41	50	0011A	.ASCII	\PASTE\
					0011F	.BLKB	1
				00000004	00120 P.ABA:	.LONG	4
				1C	00124	.BYTE	28
				04	00125	.BYTE	4
54	49	55	51		00126	.ASCII	\QUIT\
					0012A	.BLKB	2
				00000004	0012C P.ABB:	.LONG	4
				02	00130	.BYTE	2
				01	00131	.BYTE	1
				52	00132	.ASCII	\R\
					00133	.BLKB	1
				0000012C	00134 P.ABC:	.LONG	300
				18	00138	.BYTE	24
				03	00139	.BYTE	3
46	45	52			0013A	.ASCII	\REF\
					0013D	.BLKB	3
				00000004	00140 P.ABD:	.LONG	4
				0C	00144	.BYTE	12
				01	00145	.BYTE	1
				53	00146	.ASCII	\S\
					00147	.BLKB	1
				00000140	00148 P.ABE:	.LONG	320
				12	0014C	.BYTE	18
				02	0014D	.BYTE	2
				4E	0014E	.ASCII	\SN\
				00000148	00150 P.ABF:	.LONG	328
				1E	00154	.BYTE	30
				03	00155	.BYTE	3
52	48	53			00156	.ASCII	\SHR\
					00159	.BLKB	3
				00000150	0015C P.ABG:	.LONG	336
				1D	00160	.BYTE	29
				03	00161	.BYTE	3
4C	48	53			00162	.ASCII	\SHL\
					00165	.BLKB	3
				0000015C	00168 P.ABH:	.LONG	348
				0B	0016C	.BYTE	11
				03	0016D	.BYTE	3
4C	45	53			0016E	.ASCII	\SEL\
					00171	.BLKB	3
				00000168	00174 P.ABI:	.LONG	360
				06	00178	.BYTE	6
				04	00179	.BYTE	4
4C	45	53	53		0017A	.ASCII	\SSEL\
					0017E	.BLKB	2
				00000004	00180 P.ABJ:	.LONG	4
				22	00184	.BYTE	34
				02	00185	.BYTE	2
				49	00186	.ASCII	\TI\
				00000180	00188 P.ABK:	.LONG	384
				21	0018C	.BYTE	33
				02	0018D	.BYTE	2
				44	0018E	.ASCII	\TD\
				00000188	00190 P.ABL:	.LONG	392
				20	00194	.BYTE	32
				02	00195	.BYTE	2

EDT
V04


```

      43 54 00196      .ASCII \TC\
00000190 00198 P.ABM: .LONG 400
      19 0019C      .BYTE 25
      03 0019D      .BYTE 3
    50 4F 54 0019E      .ASCII \TOP\
      001A1      .BLKB 3
00000198 001A4 P.ABN: .LONG 408
      1F 001A8      .BYTE 31
      03 001A9      .BYTE 3
    42 41 54 001AA      .ASCII \TAB\
      001AD      .BLKB 3
000001A4 001B0 P.ABO: .LONG 420
      08 001B4      .BYTE 8
      04 001B5      .BYTE 4
    4A 44 41 54 001B6      .ASCII \TADJ\
      001BA      .BLKB 2
000001B0 001BC P.ABP: .LONG 432
      2C 001C0      .BYTE 44
      05 001C1      .BYTE 5
    4C 45 53 47 54 001C2      .ASCII \TGSEL\
      001C7      .BLKB 1
00000004 001C8 P.ABQ: .LONG 4
      14 001CC      .BYTE 20
      04 001CD      .BYTE 4
    57 44 4E 55 001CE      .ASCII \UNDW\
      001D2      .BLKB 2
000001C8 001D4 P.ABR: .LONG 456
      15 001D8      .BYTE 21
      04 001D9      .BYTE 4
    4C 44 4E 55 001DA      .ASCII \UNDL\
      001DE      .BLKB 2
000001D4 001E0 P.ABS: .LONG 468
      13 001E4      .BYTE 19
      04 001E5      .BYTE 4
    43 44 4E 55 001E6      .ASCII \UNDC\
      001EA      .BLKB 2
00000004 001EC P.ABT: .LONG 4
      0F 001F0      .BYTE 15
      05 001F1      .BYTE 5
    45 54 41 4C 58 001F2      .ASCII \XLATE\
      001F7      .BLKB 1
00000004 001F8 P.ABU: .LONG 4
      10 001FC      .BYTE 16
      01 001FD      .BYTE 1
      5E 001FE      .ASCII \^ \
      001FF      .BLKB 1
000000EC 000000E0 000000CC 0000007C 00000040 00000028 00200 P.ABV: .LONG 40, 64, 124, 204, 224, 236, 4, 248, 260, -
00000004 0000010C 00000004 00000104 000000F8 00000004 00218      4, 268, 4, 4, 4, 4, 276, 288, 308, 372, -
00000134 00000120 00000114 00000004 00000004 00000004 00230      444, 480, 4, 4, 492, 4, 4, 4, 4, 504
000001EC 00000004 00000004 000001E0 000001BC 00000174 00248
000001F8 00000004 00000004 00000004 00000004 00260
00000000 00278 P.ABW: .LONG 0
      00 0027C      .BYTE 0
      00 0027D      .BYTE 0
00000000 0027E      .LONG 0
      00282      .BLKB 2
00000278 00284 P.ABX: .LONG 632
```

			05	00288	.BYTE	5	
			02	00289	.BYTE	2	
	57	42	0028A	.ASCII	\BW\		
	00000284		0028C	P.ABY: .LONG	644		
		25	00290	.BYTE	37		
		02	00291	.BYTE	2		
	52	42	00292	.ASCII	\BR\		
	0000028C		00294	P.ABZ: .LONG	652		
		0B	00298	.BYTE	11		
		02	00299	.BYTE	2		
	4C	42	0029A	.ASCII	\BL\		
	00000294		0029C	P.ACA: .LONG	660		
		1B	002A0	.BYTE	27		
		04	002A1	.BYTE	4		
	52	41	50	42	002A2	.ASCII	\BPAR\
					002A6	.BLKB	2
			0000029C		002A8	P.ACB: .LONG	668
			15	002AC	.BYTE	21	
		04	002AD	.BYTE	4		
	4E	45	53	42	002AE	.ASCII	\BSEN\
					002B2	.BLKB	2
			000002A8		002B4	P.ACC: .LONG	680
			21	002B8	.BYTE	33	
			05	002B9	.BYTE	5	
	45	47	41	50	42	002BA	.ASCII
					002BF	.BLKB	1
			00000278		002C0	P.ACD: .LONG	632
			01	002C4	.BYTE	1	
			01	002C5	.BYTE	1	
			43	002C6	.ASCII	\C\	
					002C7	.BLKB	1
			00000278		002C8	P.ACE: .LONG	632
			07	002CC	.BYTE	7	
			02	002CD	.BYTE	2	
		57	45	002CE	.ASCII	\EW\	
		000002C8		002D0	P.ACF: .LONG	712	
			11	002D4	.BYTE	17	
			02	002D5	.BYTE	2	
		4C	45	002D6	.ASCII	\EL\	
		000002D0		002D8	P.ACG: .LONG	720	
			27	002DC	.BYTE	39	
			02	002DD	.BYTE	2	
		52	45	002DE	.ASCII	\ER\	
		000002D8		002E0	P.ACH: .LONG	728	
			1D	002E4	.BYTE	29	
			04	002E5	.BYTE	4	
		52	41	50	45	002E6	.ASCII
					002EA	.BLKB	2
			000002E0		002EC	P.ACI: .LONG	736
			17	002F0	.BYTE	23	
			04	002F1	.BYTE	4	
		4E	45	53	45	002F2	.ASCII
					002F6	.BLKB	2
			000002EC		002F8	P.ACJ: .LONG	748
			23	002FC	.BYTE	35	
			05	002FD	.BYTE	5	
		45	47	41	50	45	002FE
					.ASCII	\EPAGE\	

EDT\$CHMKEYWRD
V04-000

EDT\$CHMKEYWRD - look for a keyword
EDT\$\$KEY_WORD - look for a key word

C 2
16-Sep-1984 00:03:17
14-Sep-1984 12:22:37

VAX-11 Bliss-32 V4.0-742
[EDT.SRC]CHMKEYWRD.BLI;1

Page 16
(3)

DUPC-VERB=	P.AAN
DMOV-VERB=	P.AAO
DLWC-VERB=	P.AAP
DEFK-VERB=	P.AAQ
DATE-VERB=	P.AAR
D-VERBS=	P.AAS
EX-VERB=	P.AAT
E-VERBS=	P.AAU
F-VERBS=	P.AAV
H-VERBS=	P.AAW
I-VERBS=	P.AAX
K-VERBS=	P.AAY
P-VERBS=	P.AAZ
Q-VERBS=	P.ABA
R-VERB=	P.ABB
R-VERBS=	P.ABC
S-VERB=	P.ABD
SN-VERB=	P.ABE
SHR-VERB=	P.ABF
SHL-VERB=	P.ABG
SEL-VERB=	P.ABH
S-VERBS=	P.ABI
TI-VERB=	P.ABJ
TD-VERB=	P.ABK
TC-VERB=	P.ABL
TOP-VERB=	P.ABM
TAB-VERB=	P.ABN
TADJ-VERB=	P.ABO
T-VERBS=	P.ABP
UNDW-VERB=	P.ABQ
UNDL-VERB=	P.ABR
U-VERBS=	P.ABS
X-VERBS=	P.ABT
CARET-VERB=	P.ABU
VERB-TABLE=	P.ABV
END-ENTITY=	P.ABW
BW-ENT=	P.ABX
BR-ENT=	P.ABY
BL-ENT=	P.ABZ
BPAR-ENT=	P.ACA
BSER-ENT=	P.ACB
B-ENTS=	P.ACC
C-ENTS=	P.ACD
EQ-ENT=	P.ACE
EL-ENT=	P.ACF
ER-ENT=	P.ACG
EPAR-ENT=	P.ACH
ESER-ENT=	P.ACI
E-ENTS=	P.ACJ
L-ENTS=	P.ACK
N-ENTS=	P.ACL
PAR-ENT=	P.ACM
P-ENTS=	P.ACN
SR-ENT=	P.ACO
S-ENTS=	P.ACP
V-ENTS=	P.ACQ
W-ENTS=	P.ACR

				ENTITY_TABLE=	P.ACS	
				.EXTRN	EDT\$\$B_CHAR_INFO	
				.EXTRN	EDT\$\$A_CMD_END, EDT\$\$A_CMD_BUF	
				.EXTRN	EDT\$\$INTER_ERR	
			03FC 00000	.ENTRY	EDT\$\$KEY_WORD, Save R2,R3,R4,R5,R6,R7,R8,R9	0897
	59	00000000G	00 00 9E 00002	MOVAB	EDT\$\$B_CHAR_INFO, R9	
	58	00000000G	00 00 9E 00009	MOVAB	EDT\$\$A_CMD_BUF, R8	
		08	BC D4 00010	CLRL	@KEY_NUM	0958
	53		68 D0 00013	MOVL	EDT\$\$A_CMD_BUF, C_POINTER	0959
	52		83 9A 00016	MOVZBL	(C_POINTER)+, FIRST_CHAR	0960
03	6942		00 E1 00019	BBC	#0, EDT\$\$B_CHAR_INFO[FIRST_CHAR], 1\$	0965
	52		20 C2 0001E	SUBL2	#32, FIRST_CHAR	
01	01	04	AC CF 00021	CASEL	TABLE_NO, #1, #1	0973
	002A		000D 00026	.WORD	3\$-2\$,-	
					4\$-2\$	
	00000000G	00	00 FB 0002A	CALLS	#0, EDT\$\$INTER_ERR	0997
			48 11 00031	BRB	8\$	0973
	00000041	8F	52 D1 00033	CMPL	FIRST_CHAR, #65	0979
			1B 19 0003A	BLSS	5\$	
	0000005E	8F	52 D1 0003C	CMPL	FIRST_CHAR, #94	
			1B 14 00043	BGTR	6\$	
	50	FE0F	CF 9E 00045	MOVAB	VERB_TABLE, TABLE	0981
	52	BF	A2 9E 0004A	MOVAB	-65(R2), R2	0982
			1B 11 0004E	BRB	7\$	
	00000042	8F	52 D1 00050	CMPL	FIRST_CHAR, #66	0989
			73 19 00057	BLSS	13\$	
	00000057	8F	52 D1 00059	CMPL	FIRST_CHAR, #87	
			6A 14 00060	BGTR	13\$	
	50	FF42	CF 9E 00062	MOVAB	ENTITY_TABLE, TABLE	0991
	52	BE	A2 9E 00067	MOVAB	-66(R2), R2	0992
51	52		02 78 0006B	ASHL	#2, R2, TABLE_OFFSET	
	52	FBE5	CF 9E 0006F	MOVAB	ADDR_BASE, R2	0993
			6140 9F 00074	PUSHAB	(TABLE_OFFSET)[TABLE]	
50	52		9E C1 00077	ADDL3	@(SP)+, R2, TABLE_PTR	
	54	05	A0 9A 0007B	MOVZBL	5(TABLE_PTR), R4	1000
			4B 13 0007F	BEQL	13\$	
	51	06	A0 9E 00081	MOVAB	6(R0), KW_POINTER	1002
	53		68 D0 00085	MOVL	EDT\$\$A_CMD_BUF, C_POINTER	1003
	57		01 D0 00088	MOVL	#1, FOUND	1004
52	53		54 C1 0008B	ADDL3	R4, C_POINTER, R2	1006
	00000000G	00	52 D1 0008F	CMPL	R2, EDT\$\$A_CMD_END	
			29 1A 00096	BGTRU	12\$	
			55 D4 00098	CLRL	I	1010
			15 11 0009A	BRB	11\$	
	52		83 9A 0009C	MOVZBL	(C_POINTER)+, CHAR	1016
03	6942		00 E1 0009F	BBC	#0, EDT\$\$B_CHAR_INFO[CHAR], 10\$	1021
	52		20 C2 000A4	SUBL2	#32, CHAR	
	56		81 9A 000A7	MOVZBL	(KW_POINTER)+, R6	1029
	56		52 D1 000AA	CMPL	CHAR, R6	
			02 13 000AD	BEQL	11\$	
			57 D4 000AF	CLRL	FOUND	
E7	55		54 F3 000B1	AOBLEQ	R4, 1, 9\$	1010
	09		57 E9 000B5	BLBC	FOUND, 12\$	1033
	08	04	A0 9A 000B8	MOVZBL	4(TABLE_PTR), @KEY_NUM	1036
	68		54 C0 000BD	ADDL2	R4, EDT\$\$A_CMD_BUF	1040
			04 000C0	RET		1035

EDT\$CHMKEYWRD
V04-000

EDT\$CHMKEYWRD - look for a keyword
EDT\$\$KEY_WORD - look for a key word

E 2
16-Sep-1984 00:03:17
14-Sep-1984 12:22:37

VAX-11 Bliss-32 V4.0-742
[EDT.SRC]CHMKEYWRD.BLI;1

Page 18
(3)

50	54	FB93	CF	9E	000C1	12\$:	MOVAB	ADDR BASE, R4	:	1046
	54		60	C1	000C6		ADDL3	(TABLE_PTR), R4, TABLE_PTR	:	
			AF	11	000CA		BRB	8\$:	1000
				04	000CC	13\$:	RET		:	1050

; Routine Size: 205 bytes, Routine Base: _EDT\$CODE + 03A8

: 481 1051 1
: 482 1052 1 !<BLF/PAGE>

EDT\$CHMKEYWRD
V04-000

EDT\$CHMKEYWRD - look for a keyword
EDT\$\$KEY_WORD - look for a key word

F 2
16-Sep-1984 00:03:17
14-Sep-1984 12:22:37

VAX-11 Bliss-32 V4.0-742
[EDT.SRC]CHMKEYWRD.BLI;1

Page 19
(4)

: 484 1053 1 END
: 485 1054 1
: 486 1055 0 ELUDOM

! of module EDT\$CHMKEYWRD

PSECT SUMMARY

Name	Bytes	Attributes
_EDT\$CODE	1141	NOVEC,NOWRT, RD , EXE, SHR, LCL, REL, CON, PIC,ALIGN(2)

Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
-\$255\$DUA28:[EDT.SRC]EDT.L32;1	377	68	18	40	00:00.2
-\$255\$DUA28:[EDT.SRC]PSECTS.L32;1	2	1	50	7	00:00.1
-\$255\$DUA28:[EDT.SRC]TRANSLATE.L32;1	6	0	0	57	00:00.1
-\$255\$DUA28:[EDT.SRC]SUPPORTS.L32;1	2	1	50	5	00:00.1

COMMAND QUALIFIERS

: BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/NOTRACEBACK/LIS=LIS\$:CHMKEYWRD/OBJ=OBJ\$:CHMKEYWRD MSRC\$:CHMKEYWRD.BLI/UPDATE=(ENH\$:C
: HMKEYWRD)

: Size: 205 code + 936 data bytes
: Run Time: 00:29.1
: Elapsed Time: 00:33.8
: Lines/CPU Min: 2175
: Lexemes/CPU-Min: 11420
: Memory Used: 180 pages
: Compilation Complete

0131 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

0132 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

